Cryptography

To set the stage, I am not a cryptographer, cryptanalyst, nor mathematician. While I could create an encryption algorithm that I cannot break, however, I'm sure that most cryptanalysts would easily be able to break it. Therefore, I profess no cryptographic expertise at the creation or crypto-analytical levels. My exposure to encryption has been as a user with more than a passing interest in the topic and its history. I have studied cryptography and am familiar with many algorithms and with various aspects of cryptographic security as well as its long history. So, in short, my comments do not address any specific algorithm in any detail nor do they discuss the internal mathematical merits of each. I do not break codes.

Let's take a look at cryptography in general. The art and now science has been around for along time. The Greeks used the *scytale* for secret communications between their military commanders in 400 BC. It consisted of a tapered baton around which a parchment strip was spirally wrapped. A message was written down the length of the baton on the parchment. When it was unwrapped, random letters were written in between the message letters. The parchment strip was then wrapped around a currier and sent to the destination where the recipient wrapped the parchment strip around an identical baton which made the message readable. Over the intervening years, there have been many cryptographic systems of varying security.

> *The whole point of using cryptographic methods is to provide security of messages and or data in transit and at rest. If any encrypted data/message is intercepted or accessed by parties not authorized to have that access, subsequent analysis would not reveal the contents.*

In the early 20th century cryptography moved from the various mechanical and procedural systems common up to that time, into the domain of mathematics. Almost all cryptographic method used today are mathematically based. Cryptography today is actually an advanced branch of mathematics.

The first thing that I wanted to say about cryptography is that all cryptography is not created equal. There are many crypto-algorithms available to day. How do you choose? That is not a simple question. Using cryptography is really about privacy. Just because you use it does not mean that your use automatically is criminal. Some folks think that because something is secret that it is bad, criminal or illegal. It is ***NOT***. The use of cryptography provides the common person the ability to have real privacy of data – provided that strong encryption is used.

Strong encryption means that the encryption cannot be broken and has no flaws or backdoors that could be exploited by an adversary. We often hear about key bit length. That refers to the physical key size portrayed as a binary value. That value translates to the size of the key domain – the total number of unique keys possible. If an attacker can only use brute force (testing one key at a time) to derive the keys, then the number test iterations translate into time to process and determines how secure an algorithm is.

For example a popular crypt-algorithm is called *IDEA* (International Data Encryption Algorithm – a symmetric algorithm) and its key domain space is equivalent to 2 raised to the 128th power – a really big number:

340,282,366,920,938,000,000,000,000,000,000,000,000 keys

The *Fugaku* (in 2020 the world's fastest computer) could process/perform at 415.53 peta-floating-point operation-per-second (PFLOPS) – the term *peta* refers to one quadrillion or a 1 with 15 zeros after it. Another way to look at it is that it's a billion millions. In order to test any given key within a key space it takes several operations to make that test. If we disregard that fact, using the *Fugaku's* maximum processing speed to iterate through the entire key space of a 128 bit key message assuming one operation per test it would take 26 years of continuous processing to complete that task. In realty, it would be 10 or

20 times that number of operations per test. The *Fugaku* cost US$1.2 billion and there is only one of them so unless you are a determined (and rich) nation state, this capability is probably not available to any normal attacker.

This algorithm could be called computationally secure because it has no known flaws or backdoors **and** a brute force attack would take so long as to make the solution irrelevant. *IDEA* was defined and developed by James Massey and Xeujia Lai in 1991 and is now patent free so it has had plenty of time to be analysed and studied. No weakness has been found to date. No, I'm not pushing *IDEA*. However, it is an example of a strong crypto-algorithm. It is used as an integral part of Pretty Good Privacy (*PGP*) – a public key crypto-system developed by Phil Zimmermann in 1991.

It is important to understand an important philosophical point. Privacy is a human right. It is enshrined in the Universal Declaration of Human Rights (Article 12: "***No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation***") adopted by the General Assembly (Resolution 217) on 10 December 1948. On the day 48 of the 58 members voted in favour. Australia, Canada, New Zealand, the United Kingdom, and the United States among 43 other nations are amongst those who voted in favour.

This human right is not granted arbitrarily by any authority – nation state or otherwise. The right belongs to you exclusively. You and you alone get to chose whether you wish to exert this right and which crypto-algorithm to use doing it. The fact that you have chosen to use cryptography to protect your privacy does not indicate any criminal intent or activity nor should it. Now, having said all that, it is worth knowing that most nations who have ascribed to this notion of privacy will do all that they can to circumvent your privacy. They will all and any means they can to circumvent your privacy either directly or covertly. There are many examples of this kind of intrusion and a cursory search of the Internet will provide many examples.

One blatant example is a program of the National Security Agency (NSA) called *Bullrun*, revealed by Eduard Snowden. This program funded to the tune of US$250 million annually targets all encryption technologies. This entails collaboration with cryptographic vendors of software and hardware. The main objective of program *Bullrun* is to be able to break all encrypted traffic on the Internet. This is only one program – there are others. Without prim facia evidence that a crime has been or is about to be committed, no one has the right to invade your privacy (crack you encrypted messages or data). The NSA is only one example of an attacker. Some would say that they are there to protect the security interests of the US but their actions show a different picture.

If you wish to maintain you privacy, use strong encryption to protect your data (either in transit – being transmitted or at rest – stored). When an authority demands your keys, you get to choose whether or not to comply. ***Privacy should be your own choice – no exception and the use of strong data encryption is one tool that provides that choice.***

Cryptographic algorithms are generally categorized into three basic families:

1. Symmetric key algorithms
2. Asymmetric key algorithms
3. Hashing algorithms

There are other methods but these three are the most common used today. Hashing algorithms are also known as one way functions. This means that a data string that a hashing algorithm is applied against will produce a fixed length value which can be viewed as a unique fingerprint of that specific data string. If as little as a single bit was changed in the data string and then re-hashed, the fingerprint value produced would be different. This type of encryption algorithm is most commonly used for verifying passwords. Passwords are now never stored in plain text (readable as it is typed). The hash value is stored instead. When entering

a password is entered, it is hashed and then the hash value is compared to the specific account's hash value for validation. The important thing to know about a hashing algorithm is that the original data string cannot be derived from it. This is an example and not the only use of hashing algorithms.

Symmetric algorithms have only a single key (aka private key). That single key is used with the specific algorithm to encrypt the plain text (unprotected) data string producing an encrypted cipher text (protected) version of the data string. When decrypting the protected data string the same key is used with the same specific algorithm to produce a plain text (unprotected) version.

The protected version can be stored (data at rest) or sent over communication channels to a destination. While being communicated the enciphered file is protected and even if it were intercepted would be useless to the interceptor.

There are pros and cons related to this type of crypto algorithm:

1. **Pro:** symmetric algorithms are very computationally efficient. This means that they can be used to encrypt very large data strings rapidly and efficiently.
2. **Con:** The problem with symmetric algorithms is how to exchange keys in a secure way. If a key were intercepted then all that was encrypted using that key would then become vulnerable to the interceptor.

Asymmetric algorithms have two keys. These keys are mathematically related. One is designated a *public key* – one that can be published, and the other is designated the *private key* and this key must never be revealed to any – not ever.

Normally, a data string is encrypted using the *public key* belonging to the party who is to receive the protected data string. When that protected data string is received by the appropriate party they use their *private key* to decrypt the protected data string.

However, either key may be used to encrypt but the *exact opposite key* must be used to decrypt. This technique is also used to be able to sign an encrypted data string. The signature is encrypted with the private key of the sender. The receiving party can then decrypt the encrypted signature with the public key of the sender – the resulting plain text signature verifies that the message/data set was only able to be sent by the person owning the private key.

There are pros and cons related to this type of crypto algorithm:

1. **Pro:** Key management is very secure because on keys must be exchanged.
2. **Con:** Asymmetric algorithms are very computationally intensive and require a lot of computing resources.

The image below graphically demonstrates these uses.